Prepared by		Document Name		
H.Cankaya		Security Concept RNA OBC AC1		
Checked by	Released by	Release Date	Rev.	Project / Reference
		17.03.2023	0.2	

Security Concept

Abstract

This document provides an overview of the security measures integrated within the Project.

The signatures are on the original document of the related department.

Revision History

Rev.	Release Date	Prepared by	Change Description
0.1	09.02.2023	H.Cankaya	Draft Version
0.2	17.03.2023	H.Cankaya	Sections 5, 6, 7, 8 created and extended
0.3			
0.4			
0.5			
0.6			
0.7			
0.8			
1.0			
1.1			

Responsible Head of Department:	**Ulrich Nickel**

Prepared by		Document Name		
H.Cankaya		Security Concept RNA OBC AC1		
Checked by	Released by	Release Date	Rev.	Project / Reference
		17.03.2023	0.2	

Content

1. Introduction

This document provides an overview of the security measures provided in the Renault AC01 ECU.

2. Architecture

This chapter explain the architecture of the ECU including the backend components.

2.1 Architecture Overview

<todo: Architecture overview to be added>

Figure 1: Architecture Overview

<todo: System overview to be added>

Figure 2: System Overview

2.2 Communication and Network Architecture

This clause intentionally left blank.

3. Relevant Assets and Security Goals

3.1 Keys, Certificates and Secrets

	#	Key ID	Storage in ECU	Storage in Infrastructure	Dev Variant Existis	Series Variant Exists	Responsible	Key Type	Lifecycle	Plain Readable from Host	Writeable only Once	Persistent	Notes
---- Keys ----	A.1	Keypair_Delta_JTAG_Password_Encryption	Public Key stored as part of BTLD SW Code (Part is set to OTP)	Key pair stored in HSM-Server	x	x	Zerkane Salaheddine	RSA 2048 Bit					
	A.2	Keypair_Delta_SecureDiag	Public Key stored as part of BTLD SW Code (Part is set to OTP)	Key pair stored in HSM-Server	x	x	Zerkane Salaheddine / Vakarelov	RSA 2048 Bit					
	A.3	Keypair_Delta_SWSigning	Public Key stored as part of BTLD SW Code (Part is set to OTP)	Key pair stored in HSM-Server	x	x	Vakarelov / Zerkane Salaheddine	RSA 2048 Bit					
	A.4	Keypublic_Renault_SecureDiag	Public Key stored as part of BTLD SW Code (Part is set to OTP)	-	x	x	Vakarelov / Zerkane Salaheddine	RSA 2048 Bit					
	A.5	Key_OTA_Encryption	Part of SW code	No security objective for OTA_Encryption. Key will be stored in project SVN.	x	-	Cedric / Zerkane Salaheddine	AES 128 Bit					

Table 1 - Assets Storage Overview

3.2 Further Assets

This clause intntionally left blank.

4. Security Relevant HW/SW Configuration Overview

4.1 HW Configuration

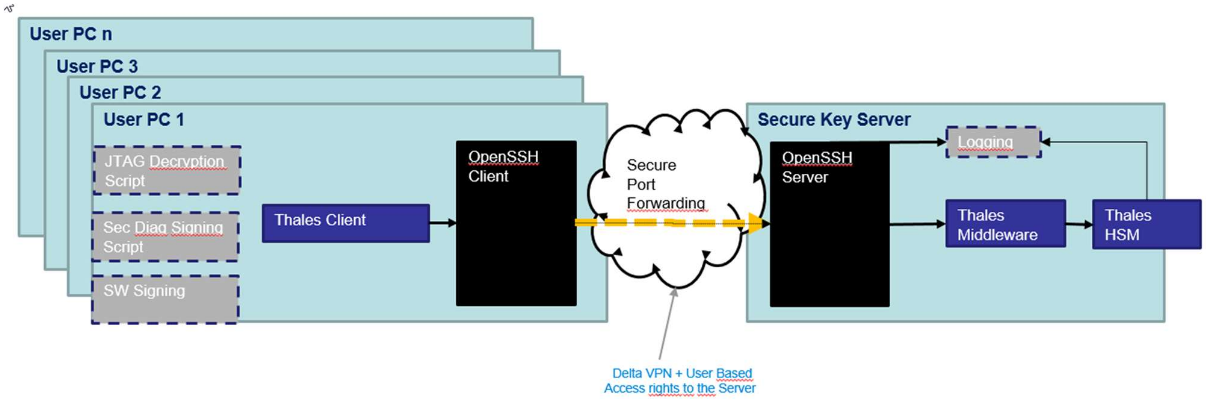
<todo: description of actions, which are needed to configure OTP and JTAG locking on HW level>

4.2 SW Configuration

This clause intentionally left blank.

5. Security Infrastructure

5.1 Secure Key Server Infrastructure Overview



5.2 Roles and Rights

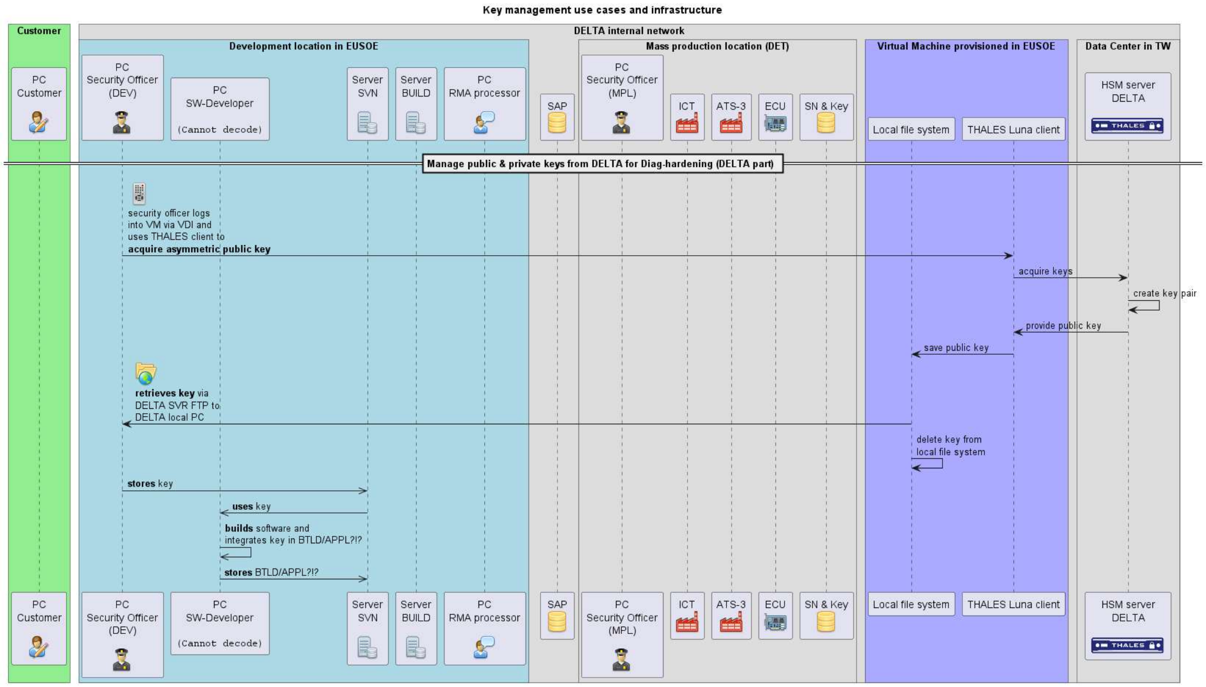
	Role						
Functionality							
	IT Admin	Security Admin	Backup Responsibles	User Role "Warranty Responsible "	User Role "Secure Diag Responsible"	User Role "SW Signing&Encryption Responsible"	Role "Build Server"
Configure Port Access for user to Secure Server	X						
Add Public Key of User to Openssh Server	X						
Initial Configuration of the HSM (Create User and Create keys)		X					
Initiate Backup Information (m of n)		X					
Recreate HSM Content			(3 of 6)				
Change Own HSM Access Password		X		X	X		
Sign Secure Diag Message (Sign with Keypair_Delta_SecureDiag via HSM)					X		
Decrypt JTAG Password (Decrypt with Keypair_Delta_JTAG via HSM)				X			
Trigger Build Server for signing and encrypting SW						X	
Sign SW (Sign with Keypair_Delta_SWSigning via HSM)							X
Encrypt SW (Encrypt with Key_OTA_Encryption via HSM)							X
Read Logs	X (on Server)	X (on HSM & on Build Server)					

5.3 Process: Store public key from RSA for Diag-hardening

The public key Keypublic_Renault_SecureDiag is received by the Delta Security Officer and stores the public key in the project SVN.

5.4 Process: Manage public & private keys from DELTA for Diag-hardening

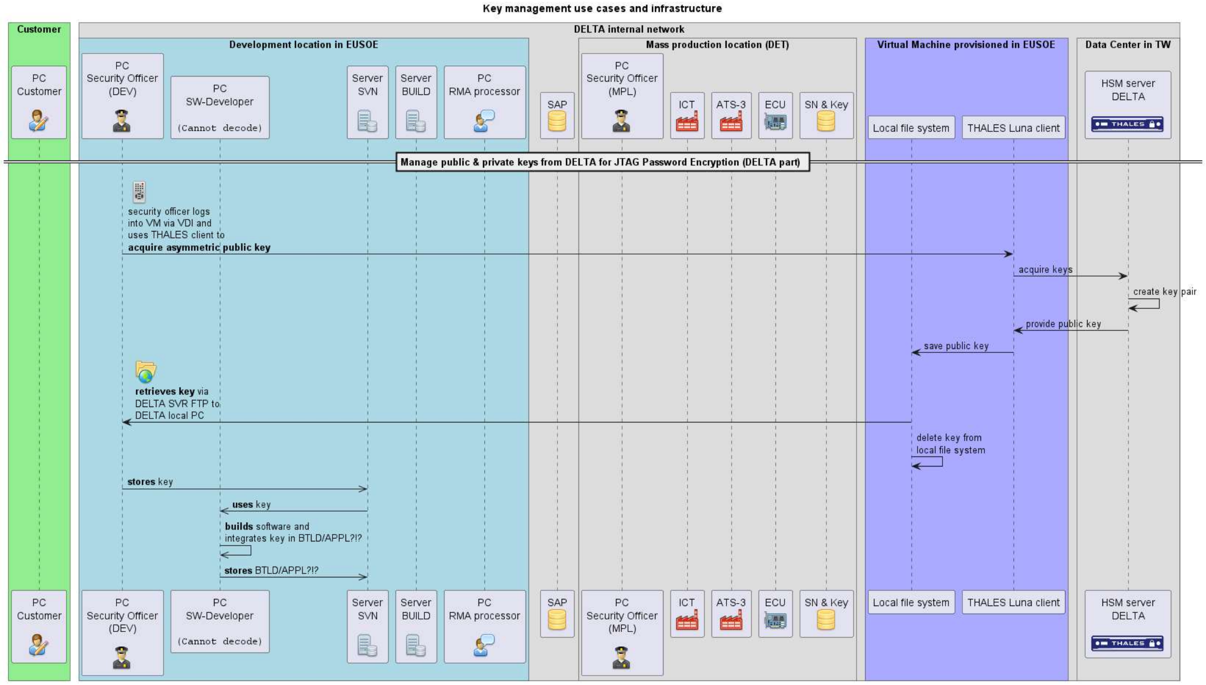
The key pair for the generation and verification of the security access response is created by Delta in the secure key server. The private key does not leave the secure key server. The public key is part of the bootloader image.


Key management use cases and infrastructure
Customer
DELTA internal network
Development location in EUSOE
Mass production location (DET)
Virtual Machine provisioned in EUSOE
Data Center in TW
PC Customer
PC Security Officer (DEV)
PC SW-Developer (Cannot decode)
Server SVN
Server BUILD
PC RMA processor
SAP
PC Security Officer (MPL)
ICT
ATS-3
ECU
SN & Key
Local file system
THALES Luna client
HSM server DELTA
THALES
Manage public & private keys from DELTA for Diag-hardening (DELTA part)
security officer logs into VM via VDI and uses THALES client to acquire asymmetric public key
acquire keys
create key pair
provide public key
save public key
retrieves key via DELTA SVR FTP to DELTA local PC
delete key from local file system
stores key
uses key
builds software and integrates key in BTLD/APPL?!?
stores BTLD/APPL?!?

5.5 Process: Manage keys for JTAG encryption

The JTAG password encryption key pair is created within the secure key server. The public key is stored in the bootloader code and is part of the bootloader image.

For more information on the use of the public and private key please refer to section 6.3.


Key management use cases and infrastructure
Customer
DELTA internal network
Development location in EUSOE
Mass production location (DET)
Virtual Machine provisioned in EUSOE
Data Center in TW
PC Customer
PC Security Officer (DEV)
PC SW-Developer (Cannot decode)
Server SVN
Server BUILD
PC RMA processor
SAP
PC Security Officer (MPL)
ICT
ATS-3
ECU
SN & Key
Local file system
THALES Luna client
HSM server DELTA
THALES
Manage public & private keys from DELTA for JTAG Password Encryption (DELTA part)
security officer logs into VM via VDI and uses THALES client to acquire asymmetric public key
acquire keys
create key pair
provide public key
save public key
retrieves key via DELTA SVR FTP to DELTA local PC
delete key from local file system
stores key
uses key
builds software and integrates key in BTLD/APPL?!?
stores BTLD/APPL?!?

5.6 Process: Manage keys for code signing

The SW Code Signing key pair is created within the secure key server. The public key is read out from the HSM server by the build server and added to bootloader image during the build process. The build server has secure access to the secure key server and can sign SW images with the code signing key.

5.7 Process: Provision of the software by the development to the production

The software development team is located at DES and will provide the final built software via DELTA's internal IT network to the mass production location in DET as part of the BOM. The BOM is stored in the internal SAP system. The final software mainly consists out of three software package, the bootloader (BTLD), the application software for prodution (APPL_production) and series application software (APPL_series). DET will retrieve the software and provide it to the specific production stations, e.g. ICT or ATS. At ICT or ATS, the software will be flashed into the electornic control unit (ECU):

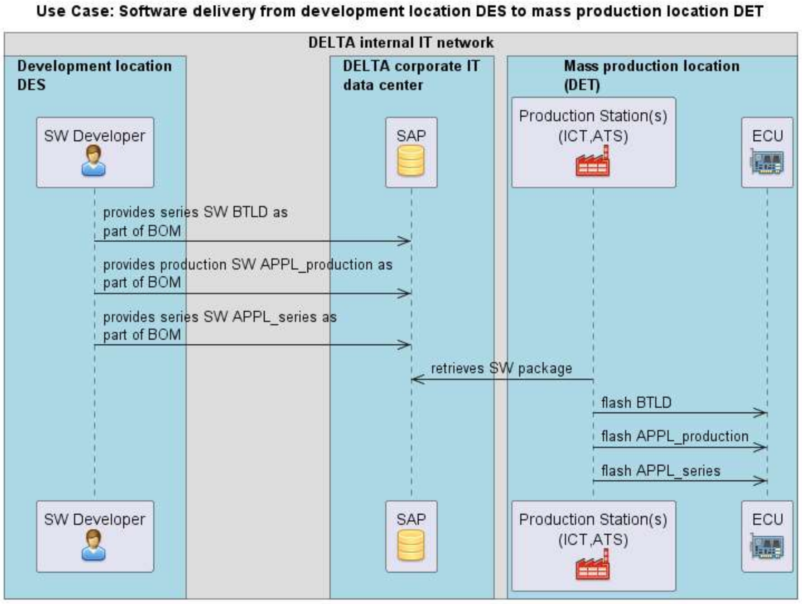

Use Case: Software delivery from development location DES to mass production location DET
DELTA internal IT network
Development location DES
DELTA corporate IT data center
Mass production location (DET)
SW Developer
SAP
Production Station(s) (ICT,ATS)
ECU
provides series SW BTLD as part of BOM
provides production SW APPL_production as part of BOM
provides series SW APPL_series as part of BOM
retrieves SW package
flash BTLD
flash APPL_production
flash APPL_series
SW Developer
SAP
Production Station(s) (ICT,ATS)
ECU

Figure 3: Provision of the software by the development to the production, [use_case_sw_to_mpl.png].

5.8 Setting up the Secure Key Server (HSM-Server)

This is performed by DELTA corporate IT.

5.9 Production Server

For some processes a local server for the production site is needed.

The Production Server is used for following processes

- Store Encrypted JTAG Password
- Read Encrypted JTAG Password for specific Serial ID's

6. Security Measures and Processes

6.1 RNG Creation in the ECU

The ECU does not have a hardware security module for providing trustable entropy for a random number generator. The current approach to create random numbers is following.

- The random number is created with the PRNG algorithm provided by the Vector Crypto Module Crypto_30_LibCV.
- The random number is created according to FIPS 186-2.
- The PRNG is seeded at every random number generation with the current timer value XOR serial number of the device from the HW timer. The HW timer is running in the same frequency as the processor and has 32 bits length. The serial number is a constant device specific 20 byte long data.

6.2 Secure XCP Access

XCP is active under following conditions

- While the ECU is in virgin mode.
- When XCP is activated with the service “2E 72 10 - Delta_Internal_XCP_Activation_Write”

The service “2E 72 10 - Delta_Internal_XCP_Activation_Write” is only accessible in the security access level 3. XCP is only active until the device goes into sleep mode or a reset.

6.3 Secure JTAG Access

6.3.1 Secure JTAG Access – Production Process

The JTAG interface is locked during the production at Delta. The JTAG is locked by the processor UCB[1] configuration and the restriction is assured by the HW measures of the processor. JTAG can be unlocked by a 256 bit password. The password is created randomly internally in the ECU during the production process at Delta, see clause 6.1. The password is then encrypted with public key of Keypair_Delta_JTAG_Passwod_Encryption and sent to the production station. The production station stores the encrypted JTAG password in the local secure production server.

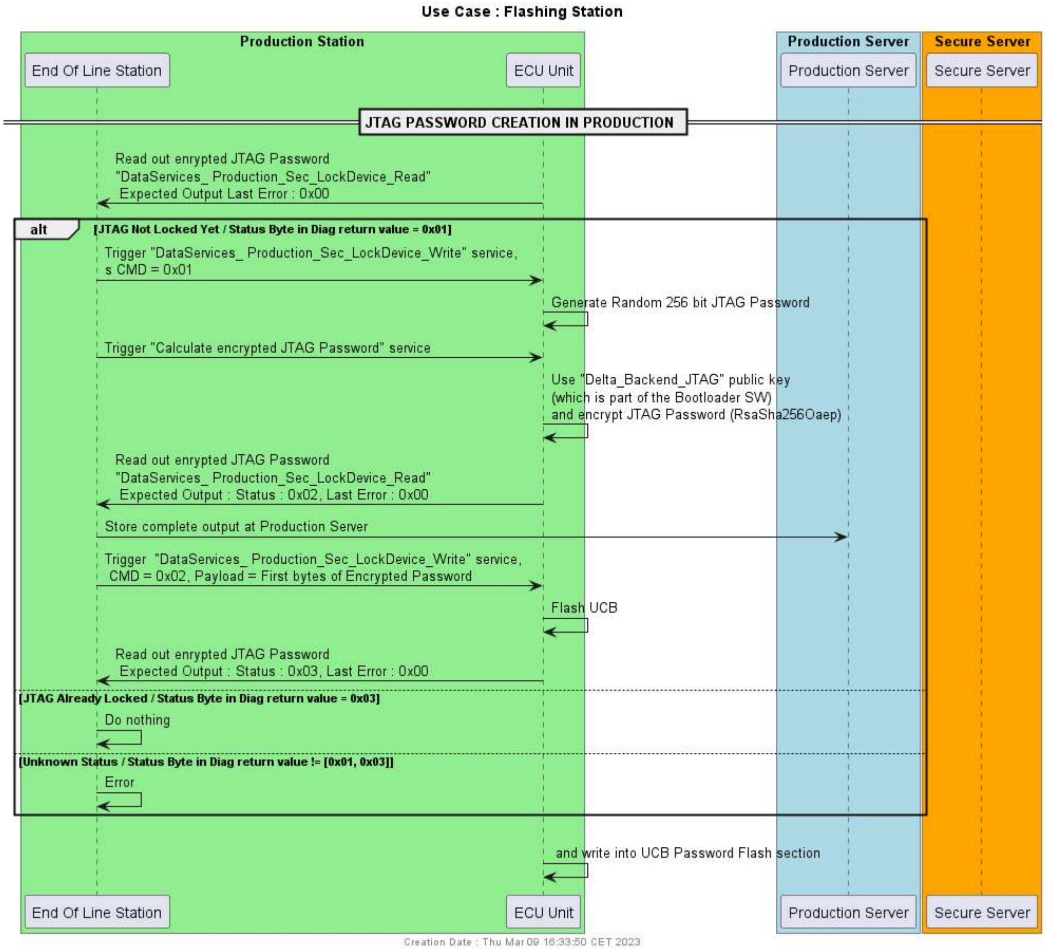

Use Case : Flashing Station
Production Station
Production Server
Secure Server
End Of Line Station
ECU Unit
Production Server
Secure Server
JTAG PASSWORD CREATION IN PRODUCTION
Read out enrypted JTAG Password
"DataServices_ Production_Sec_LockDevice_Read"
Expected Output Last Error : 0x00
alt
[JTAG Not Locked Yet / Status Byte in Diag return value = 0x01]
Trigger "DataServices_ Production_Sec_LockDevice_Write" service,
s CMD = 0x01
Generate Random 256 bit JTAG Password
Trigger "Calculate encrypted JTAG Password" service
Use "Delta_Backend_JTAG" public key
(which is part of the Bootloader SW)
and encrypt JTAG Password (RsaSha256Oaep)
Read out enrypted JTAG Password
"DataServices_ Production_Sec_LockDevice_Read"
Expected Output : Status : 0x02, Last Error : 0x00
Store complete output at Production Server
Trigger "DataServices_ Production_Sec_LockDevice_Write" service,
CMD = 0x02, Payload = First bytes of Encrypted Password
Flash UCB
Read out enrypted JTAG Password
Expected Output : Status : 0x03, Last Error : 0x00
[JTAG Already Locked / Status Byte in Diag return value = 0x03]
Do nothing
[Unknown Status / Status Byte in Diag return value != [0x01, 0x03]]
Error
and write into UCB Password Flash section
End Of Line Station
ECU Unit
Production Server
Secure Server
Creation Date : Thu Mar 09 16:33:50 CET 2023

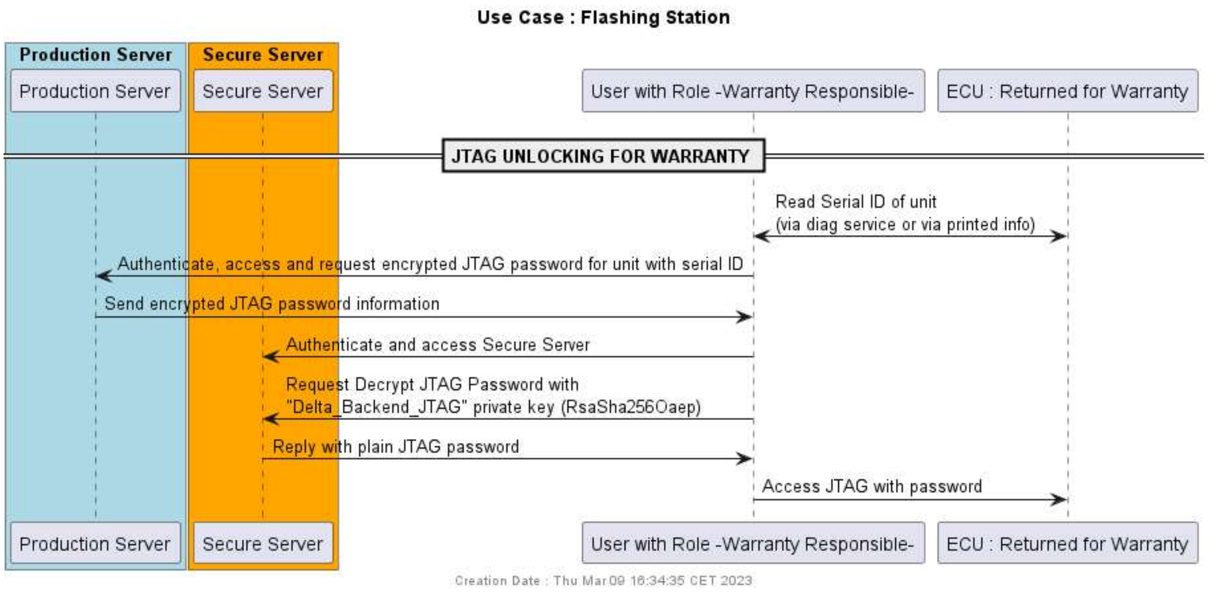
Figure 4 - JTAG Locking at Production

[1] The password in the UCB register cannot be read out anymore once JTAG locking is activated.

6.3.2 Access JTAG Password in case of Warranty

The JTAG password can be recovered by the security officer by performing the following steps:

- Read the unique ID of the device
- Retrieve the encrypted JTAG password from the production server
- Decrypt the JTAG password with the help of the HSM server


Use Case : Flashing Station
Production Server
Secure Server
Production Server
Secure Server
User with Role -Warranty Responsible-
ECU : Returned for Warranty
JTAG UNLOCKING FOR WARRANTY
Read Serial ID of unit
(via diag service or via printed info)
Authenticate, access and request encrypted JTAG password for unit with serial ID
Send encrypted JTAG password information
Authenticate and access Secure Server
Request Decrypt JTAG Password with
"Delta_Backend_JTAG" private key (RsaSha256Oaep)
Reply with plain JTAG password
Access JTAG with password
Production Server
Secure Server
User with Role -Warranty Responsible-
ECU : Returned for Warranty
Creation Date : Thu Mar 09 16:34:35 CET 2023

6.3.3 JTAG Unlocking

The JTAG interface can be unlocked with the correct password. Precondition is that the password for the specific device has to be retrieved as described in section 6.3.2. The password has to be added into the Debuger tool, and unlock the interface while the debuger connects to the JTAG interface. Please refer to the user manual of the debugger tool for further details.

6.4 Securing Diagnostic Services

6.4.1 Secure Diagnostic – General Information

Some diagnostic services can only be executed once a specific security level is activated. The UDS service "Security Access" (0x27) is used to activate the security levels.

2 set of security level is defined

- Security Access OEM Level 1 services $27 01, $27 02
- Security Access Supplier Level 3 services $27 03, $27 04

The following sections define which diagnostic services are protected by the security access.

6.4.2 Security Access for Delta Services (Security Access Level 3)

The following diagnostic services are only accessible in the DELTA diagnostic session:

- $2E : Delta Internal Service using WriteDataByIdentifier $2E
  - Enable the XCP : 2E 72 10 - Delta_Internal_XCP_Activation_Write
  - 2E 70 00 - Delta_Internal_Access_Write
  - 2E 72 11 - Delta_Internal_Debug_Activation_Write
  - 2E 72 13 - Delta_Internal_Class_E_Fault_Write
  - 2E 72 14 - Delta_Internal_ECU_Serial_Number_Write
  - 2E 73 10 - Delta_Internal_VMSparePartNumber_R_Write
  - 2E 73 11 - Delta_Internal_VMSparePartNumber_N_Write
  - 2E 73 12 - Delta_Internal_VMEcuHardwareNumber_Write

6.4.3 Security Access for OEM Services (Security Access Level 1)

The following diagnostic services are only accessible in the OEM diagnostic session:

- $2E : All Write DataByIdentifier service with DiagTool
- $31 : All Routine control service with DiagTool
- $10 : Programming Request with DiagTool
- $11 01: ECU Hard Reset via DiagTool

6.5 Software Privacy and Integrity Protection

6.5.1 SW Signing and Verificaiton Process

The software signing process is described in clause 5.6.

The SW update and verification process in the ECU is described as following

1. ECU is running in Application
2. Security Access in OEM session
3. Run diag service to switch to BTLD mode (only accessible in OEM session)
4. Receive flashing request and header containing information of SW package and the signature
5. Store signature in NvM
6. Start Download Request : Erase Flash
7. Set existing Application to invalid
8. Receive SW chunks and flash them in the Application Flash area
9. Verify Signature of flashed SW
10. Set Application to Valid if Signature is valid
11. Reset
12. Start new Application

6.5.2 SW Decryption process in the ECU

There is no security objective for SW encryption/decryption. Nevertheless SW encryption will be implemented due to compatibilitiy issues. Therefor the OTA encryption key is not specifically protected in the infrastructure or in the ECU.

Further information on the encryption/decryption process in the ECU is not included in this document, as it is not security relevant.

7. Key / Bootloader Update Procedures

The bootloader image incorporates the following assets.

- Keypair_Delta_JTAG
- Keypair_Delta_SecureDiag
- Keypair_Delta_SWSigning
- Keypublic_Renault_SecureDiag
- Key_OTA_Encryption

Until the series keys are created in the secure key server at Delta, we will proceed using development keys.

An Update of the Bootloader is only planned for updating from a development bootloader (containing development keys) to a series bootloader (containing series keys). A bootloader updater for the series devices is not planned, therefor a downgrading from series to development is not planned.

Once the Bootloader is updated from development to series, all keys can be updated to the series key.

8. Delta Internal Security Diagnostic Services

8.1 DataServices_ Production_Sec_LockDevice_Write:

Start ->

DID :

Preconditions :

- ECU in Virgin Mode
- is running in Bootloader

Byte	Meaning	Value
0	Command	0: Do nothing 1: Generate Jtag Password 2: Flash Jtag Password and Lock Device
1..12	Payload	In case of CMD = 0x01 -> Payload = magicword = 0x11 0xDD 0x34 0x91 0x00 0x00 … In case of CMD = 0x02 -> Payload = First 12 bytes of Encrypted Password (see DataServices_ Production_Sec_LockDevice_Read)

8.2 DataServices_ Production_Sec_LockDevice_Read:

Read ->

DID :

Preconditions :

- ECU in Virgin Mode
- ECU is running in Bootloader

Byte	Name	Meaning	Value
0	Status	States the status of the JTAG locking state	0x01 : Not Locked 0x02 : Password generated and encrypted, Not flashed yet in UCB, device not locked yet 0x03 : JTAG Locked Else : Not Known
1	Last Error	Error from last Call of RoutineControl_ Production_Sec_LockDevice_Start	0x00 : No Error 0x01 : Password generation error 0x02 : Encryption Error 0x03 : Flashing JTAG Password Error 0x04 : Command Content Wrong 0x05 : No valid Serial ID (e.g. still default value) 0x06 : Device already locked, no passwprd creation or flashing possible 0xFF : General Error
2 – 21	Serial Number	20 Bytes Serial ID of Device	
22-29	Pub Key	First 8 bytes of the Public key used to encrypt the Password	

30-305	Encrypted Password	256 byte encrypted SerialID + Password	If not existing = 0x00 …. 0x00

8.3 DataServices_ Production_Sec_Write_Fota_SubKey_2:

Start ->

DID :

Preconditions :

- ECU in Virgin Mode
- Is running in Bootloader

Byte	Meaning	Value
0	Command	0: Do nothing 1: Write Encryped Fota_SubKey_2 Else : Do Nothing
1..16	16 bytes of Data of encrypted Fota_Subkey_2	

8.4 DataServices_ Production_Sec_Read_Key_Attestation:

Start ->

DID :

Preconditions :

- is running in Application or Bootloader

Byte	Meaning	Value
0..7	Fingerprint Keypublic_Delta_JTAG	First 8 bytes of SHA256 of the Keypublic_Delta_JTAG
7..15	Fingerprint Keypublic_Delta_SecureDiag	First 8 bytes of SHA256 of the Keypublic_Delta_SecureDiag
16..23	Fingerprint Keypublic_Renault_SecureDiag	First 8 bytes of SHA256 of the Keypublic_Renault_SecureDiag
24..31	Fingerprint Keypublic_Delta_SWSigning	First 8 bytes of SHA256 of the Keypublic_Delta_SWSigning
32..39	Fingerprint Key_OTA_Encryption	First 8 bytes of SHA256 of the Key_OTA_Encryption

9. Abbreviations and glossary

Abbreviation or Term	Description
ATS	Automated Test System
BOM	Bill of Material
DES	Development location in DELTA Soest, Germany
DET	Mass production location in DELTA Thailand
ECU	Electronic Control Unit
ICT	Integrated Circuit Tester